UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/810,716	03/16/2001	Hiang-Swee Chiang	**GP-0002	2184

23377 7590 12/13/2007

WOODCOCK WASHBURN LLP
CIRA CENTRE, 12TH FLOOR
2929 ARCH STREET
PHILADELPHIA, PA 19104-2891

EXAMINER	
WOOD, WILLIAM H	

ART UNIT	PAPER NUMBER
2193	

MAIL DATE	DELIVERY MODE
12/13/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/810,716
Filing Date: March 16, 2001
Appellant(s): CHIANG, HIANG-SWEE

John E. McGlynn
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 04 September 2007 appealing from the Office action mailed 17 May 2006.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is substantially correct. The changes are as follows:

GROUNDS OF REJECTION NOT ON REVIEW

The following grounds of rejection have not been withdrawn by the examiner, but they are not under review on appeal because they have not been presented for review in the appellant's brief. Claims 14, 16-17, 21, 38, 40-41, 58, 60-62, 71 and 73-74 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lau** (USPN 5,987,247) in view of **Lindhorst** et al. (USPN 6,337,696) in view of **Quaeler-Bock** et al. (USPN 6,023,271) and in further view of **APA** (Applicant Admitted Prior Art).

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

5,987,247	LAU	11-1999
6,337,696	LINDHORST et al.	1-2002
6,023,271	QUAELER-BOCK et al.	2-2000

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 2, 4-13, 15, 18-20, 22-29, 31-37, 39, 42-50, 52-57, 59, 63, 64, 66-70, 72, 75-78 and 162-169 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lau** (USPN 5,987,247) in view of **Lindhorst** et al. (USPN 6,337,696) in further view of **Quaeler-Bock** et al. (USPN 6,023,271).

Claim 2

Lau disclosed a method of generating computer code for a application *(column 5, lines 33-40)*, comprising:

generating an application framework code and an event handler skeleton *(column 6, lines 24-29; column 5, lines 33-40; column 13, lines 28-44; column 5, lines 33-40)*,

receiving application business logic objects *(column 6, lines 24-29)*;

receiving methods *(column 13, lines 28-44)*;

organizing the application framework code, the application business logic objects and the event handler methods into application source code *(column 4, lines 25-27, linking and compiling, along with building and preparing the code)*;

Lau did not explicitly state generating code for a *web* application. **Lindhorst** demonstrated that it was known at the time of invention to generate code for web applications (column 2, lines 11-19; column 3, lines 1-4; and column 4, lines 10-16) using, among other elements, a graphical interface input file (column 11, lines 37-40). It would have been obvious to one of ordinary skill in the art at the time of invention to implement the application framework generation system of **Lau** with graphical design input for the web as found in **Lindhorst**'s teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide less technical and thus easier methods, such as frameworks and automatic code generation, for average users to program in various known environments, like the web (**Lindhorst**: column 3, lines 37-45; **Lau**: column 2, lines 43-47).

Lau did not explicitly state *receiving event handler methods; and wherein generating an event handler skeleton further comprises: parsing at least one input file (**Lindhorst**: column 11, lines 37-40); reviewing the parsed input file for a tag type, an attribute name and an attribute value (**Lindhorst**: column 13, lines 22-64); and determining an event handler method based on the tag type, the attribute name and the attribute value (**Lindhorst**: column 13, lines 22-64).* **Lindhorst** demonstrated that it was known at the time of invention to provide event handler methods (column 3, lines 16-20; column 11, line 66 to column 12, line 17; figure 6) and generating/producing them in the manner cited

above. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the code generation system of **Lau** with provided event handler methods as found in **Lindhorst**'s teaching. Further, it would have been obvious to one of ordinary skill in the art at the time of invention to implement the generation of event handler methods as described above through **Lindhorst**. This implementation would have been obvious because one of ordinary skill in the art would be motivated to free a user from being required to know complex technical details of programming, thus making it easier (**Lindhorst**: column 3, lines 38-45).

Lau and **Lindhorst** did not explicitly state *compiling/binding the web application source code with input files at runtime or receive GUI input files.* **Quaeler-Bock** demonstrated it was known at the time of invention to bind at runtime GUI components (input files) to business objects (source code) (column 3, lines 21-25) and thus receiving GUI input files. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the code development system of **Lau** and **Lindhorst** with run-time binding of various files/inputs as suggested by **Quaeler-Bock**'s teachings. This implementation would have been obvious because one of ordinary skill in the art would be motivated to reduce error-prone operations during code development (**Quaeler-Bock**: column 3, lines 5-10). Finally, runtime interpretation/binding indicates allowing updated/modified files up to runtime (ie. "synchronization").

Claim 4

Lau and **Lindhorst** disclosed the method of claim 2, wherein the web application source code is generated in an object-oriented programming language *(**Lau**: column 6, line 34).*

Claim 5

Lau and **Lindhorst** disclosed the method of claim 4, wherein the object-oriented programming language is Java *(column 6, line 34).*

Claim 6

Lau and **Lindhorst** disclosed the method of claim 4, wherein the object-oriented programming language is C++ *(column 3, line 65).*

Claim 7

Lau and **Lindhorst** disclosed the method of claim 2, further comprising determining if the application framework code is available for the web application *(**Lau**: column 6, lines 21-22; and column 5, lines 46-50; must determine if saved framework exists for future editing/ changing).*

Claim 8

Lau and **Lindhorst** disclosed the method of claim 2, further comprising generating a business logic foundation code *(**Lau**: column 6, lines 24-29).*

Claim 9

Lau and **Lindhorst** disclosed the method of claim 2, further comprising generating a graphical user interface code *(**Lau**: column 5, line 39).*

Claim 10

Lau and **Lindhorst** disclosed the method of claim 9, wherein generating a graphical user interface code is based on the input files *(**Lau**: column 5, lines 33-39; design; column 4, lines 11-28).*

Claim 11

Lau and **Lindhorst** disclosed the method of claim 2, wherein generating an event handler skeleton is based on the input files *(**Lau**: column 5, lines 33-39; design; column 4, lines 11-28).*

Claim 12

Lau and **Lindhorst** disclosed the method of claim 2, further comprising compiling the web application source code *(**Lau**: column 4, lines 25-27).*

Claim 13

Lau and **Lindhorst** did not explicitly state the method of claim 2, further comprising interpreting the web application source code. **Lau** demonstrated that it was known at the time of invention to implement using JAVA (column 6, lines 34). Official Notice is taken that Java technology is known to include a interpretation system. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the code development system of **Lau** and **Lindhorst** with interpreting code as suggested by JAVA's teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide a system of easy programmability (interpretation can be changed quickly on the fly, especially useful in system testing).

Claim 15

Lau and **Lindhorst** disclosed the method of claim 2, wherein the input files are in HTML format ***(Lindhorst**: column 11, lines 37-40).*

Claim 18

Lau and **Lindhorst** disclosed the method of claim 2, further comprising receiving modified input files *(see motivation under claim 2; runtime interpretation/binding indicates allowing updated/modified files up to runtime).*

Claim 19

Lau and **Lindhorst** did not explicitly state the method of claim 18, further comprising compiling the modified input files at runtime. **Lau** demonstrated that it was known at the time of invention to implement using JAVA (column 6, lines 34). Official Notice is taken that Java technology is known to include a just-in-time compiling system (in other words compiling at run time). It would have been obvious to one of ordinary skill in the art at the time of invention to implement the framework development system of **Lau** and **Lindhorst** with run-time compiling as suggested by JAVA's teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to allow for changes/improvements right up until actual use of code (see claim 18).

Claim 20

Lau and **Lindhorst** disclosed the method of claim 19, further comprising binding the web application source code with the modified input files at runtime (see claim 2 above).

Claim 22

Lau and **Lindhorst** did not explicitly state the method of claim 18, further comprising interpreting the modified input files at runtime. **Lau** demonstrated that it was known at the time of invention to implement using JAVA (column 6,

lines 34). Official Notice is taken that Java technology is known to include a interpretation system. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the development system of **Lau** and **Lindhorst** with interpreting code as suggested by JAVA's teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide a system of easy programmability (interpretation can be changed quickly on the fly, especially useful in system testing).

Claim 23

Lau and **Lindhorst** disclosed the method of claim 22, further comprising binding the web application source code with the interpreted modified input files at runtime *(see claim 2 and 22; further binding required in order for code to work correctly).*

Claim 24

Lau and **Lindhorst** disclosed the method of claim 2, further comprising generating application runtime properties ***(Lau**: column 5, lines 39-40; attributes at least).*

Claim 25

Lau and **Lindhorst** did not explicitly state the method of claim 2, further comprising generating application SQL statements. **Lau** demonstrated that it was known at the time of invention to utilize database management systems in business logic (column 8, lines 16-25). Official Notice is taken that SQL was known at the time of invention. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the code framework system of **Lau** with generating SQL as well. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide **Lau**'s system with the ability to communicate with as many differing systems/environments as possible and thus increasing flexibility and usability.

Claim 26

Lau and **Lindhorst** disclosed the method of claim 2, wherein the application framework code comprises an application object and a servlet web application framework object *(column 5, lines 15-19)*.

Claims 27-29, 31-37, 39, 42-50, 52-57, 59, 63, 64, 66-70, 72, 75-78 and 162-166

The limitations of system claims 27-29, 31-37, 39, 42-50, 52-57, 59, 63, 64, 66-70, 72, 75-78 and 162-166 correspond to the limitations of method claims 2, 4-13, 15, 18-20 and 22-26 and as such are rejected in the same manner.

Claim 167

Lau, **Lindhorst** and **Quaeler-Bock** disclosed the method of claim 1, further comprising:

determining if an application framework code is available for the web application *(column 6, lines 21-22; and column 5, lines 46-50; must determine if saved framework exists for future editing/changing)*; and

if the application framework is not available, then generating the application framework code *(column 5, lines 33-40; requiring generation if no saved information is present).*

Claims 168-169

The limitations of claims 168-169 are substantially the same as for claim 167 and as such are rejected in the same manner.

Claims 14, 16-17, 21, 38, 40-41, 58, 60-62, 71 and 73-74 are rejected under 35 U.S.C. 103(a) as being unpatentable over **Lau** (USPN 5,987,247) in view of **Lindhorst** et al. (USPN 6,337,696) in view of **Quaeler-Bock** et al. (USPN 6,023,271) and in further view of **APA** (Applicant Admitted Prior Art).

Claim 14, 16-17

Lau and **Lindhorst** did not explicitly state the method of claim 2, wherein the input files are in XML, cHTML or WML format. **APA** demonstrated that it was known at the time of invention to utilize XML and WML (page 3, lines 13-14). Official Notice is take that cHTML was known at the time of invention. It would have been obvious to one of ordinary skill in the art at the time of invention to implement the input files of **Lau** and **Lindhorst** with the above formats as found in **APA**'s teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide as many formats as possible in order to be of use to the largest community of developers possible and thus increase usefulness of the system.

Claim 21

Lau and **Lindhorst** did not explicitly state the method of claim 20, wherein the modified input files are compiled into DOM objects at runtime *(APA: page 3, lines 14-16)*. **APA** demonstrated that it was known at the time of invention to compile mark up language files into DOM (page 3, lines 14-16). It would have been obvious to one of ordinary skill in the art at the time of invention to implement the code development system of **Lau** and **Lindhorst** with DOM compilation as found in **APA**'s teaching. This implementation would have been obvious because one of ordinary skill in the art would be motivated to provide a easily handled structure for development (**APA**: page 3, lines 16-22).

Claims 38, 40-41, 58, 60-62, 71 and 73-74

The limitations of claims 38, 40-41, 58, 60-62, 71 and 73-74 are substantially the same as for claims 14, 16-17 and 21 and as such are rejected in the same manner.

(10) Response to Argument

The cited prior art in question, **Lindhorst** et al. (USPN 6,337,696), discloses a system and method for generation and editing of event handlers using a variety of graphical user interfaces/windows/panes (column 2, lines 40-50; column 2, lines 62-67) and automatic generation of event handling (column 3, lines 1-29).

Appellant's arguments filed 04 September 2007 have been fully considered but they are not persuasive. Appellant argues the cited prior art fails to disclose "generating an event handler skeleton compris[ing] ... determining an event handler method based on one or more of the tag type, the attribute name and attribute value" (Brief: page 10, section A). This argument is not found to be persuasive in view of the cited prior art. The claimed terminology is of sufficient breadth that "determining ... based on" is proven so long as the elements are present at any point in the process.

First, though the user does have a role in the cited prior art, the system of **Lindhorst** is very much involved in the "determining an event handler

method based on one or more of the tag type, the attribute name and the attribute value". This is clear from "outputs events to the event pane corresponding to events that are associated with scriptable HTML tags" (column 13, lines 22-24) and the subsequent Table 2 (column 13, lines 29-65).

Second, under the broadest reasonable interpretation an event handler *is* determined by "one or more of the tag type, the attribute name and the attribute value". Refer to Table 2 (**Lindhorst** column 13, lines 30-65), showing "events" and "methods" related to specific "HTML tags". A user is involved in the process but that involvement is "based on" certain tags having certain/appropriate events and methods. Further note, Table 2's "HTML tag" demonstrates tag type (for example scriptable tags; column 13, line 24), attribute name (for example FORM) and attribute value (for example "button"). Further, the system generates an output including event handler methods (column 3, lines 16-20; column 11, line 66 to column 12, line 17; figure 6, corresponding text found at column 17, line 22 to column 20, line 31). Focusing on Table 2 of **Linhorst**, support is clear for event handler methods to be determined based on the table itself, the user, and the wizard program producing the final event handling (column 13, lines 30-65; interestingly column 13, line 65 to column 14, line 7 and column 14, lines 46-55). The cited prior art discloses the broadest reasonable interpretation of the fairly broad claims.

Third, to the extent that a user's selection may have an impact on the overall process, the terminology "determining ... based on" has a broad meaning. The phrase "based on" does not confer that a user must choose "solely based on" nor does it even indicate that a user must select a course of action in which the "tag type, attribute name and attribute value" have the most important, final or direct impact. The current claim language requires "determining ... based on one or more of the tag type, the attribute name and the attribute value". A determination must therefore simply take into account or have within the process "one or more of the tag type, the attribute name and the attribute value". This is clearly evident in that above cited portions of **Lindhorst** disclose "tag type, attribute name and attribute value" (column 13, lines 22-67, specifically column 13, lines 22-24 and lines 65-67) as part of a larger process to determine event handler methods, whereby a user links based on system output to the event pane and the action pane (column 3, lines 16-20; column 11, line 66 to column 12, line 17). This output includes suggestions as indicated in Table 2 (column 13, lines 31-64), which indicate "tag type, attribute name and attribute value". The end determination is "based on" the "tag type, attribute name and attribute value" by virtue of their place in the process. The claim language provides nothing other than this simple meaning.

Finally, it is clear under the broadest reasonable interpretation of the claim language, that "determining ... based on" provides little meaning other

than the overall determination process must involve in some undefined way a reference to "event handler methods" being produced downstream from a consideration that involves "one or more of the tag type, the attribute name and the attribute value". This is clearly demonstrated by **Lindhorst** as explained. Therefore, the rejections have been maintained.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

William H. Wood

WILLIAM WOOD
PRIMARY EXAMINER

Conferees:

Meng Ai An

MENG-AI T AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Kakali Chaki
TC 2100 AAS